PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 9/44, 9/445

A1

(11) International Publication Number: WO 98/27487

(43) International Publication Date: 25 June 1998 (25.06.98)

(21) International Application Number: PCT/US97/20396

(22) International Filing Date: 7 November 1997 (07.11.97)

(30) Priority Data:
08/769,634 18 December 1996 (18.12.96) US

(71) Applicant: DSC TELECOM L.P. [US/US]; 1000 Coit Road, Plano, TX 75075 (US).

(72) Inventors: CARRIER, David, F., III; 4521 Charlemagne Drive, Plano, TX 75093 (US). GILLESPIE, R., John, K.; 3806 Maywood Drive, Carrollton, TX 75007 (US). LUI, Janet, Kwai, Fun; 7780 McCallum Boulevard #22205, Dallas, TX 75252 (US). WEEKS, Donald, L., Jr.; 712 Melrose Drive, Richardson, TX 75080 (US).

(74) Agent: JEANG, Wei, Wei; Baker & Botts, L.L.P., 2001 Ross Avenue, Dallas, TX 75201-2980 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: SOFTWARE RELEASE MEDIA DOWNLOAD SYSTEM AND METHOD

(57) Abstract

Source modules (160) are checked into a version control subsystem (12) and check-in data are sent to the file release control system (10). System (10) provides for the selection of check-in data for attaching to a release form (162), the entry of additional information, such as a problem report number associated with the source module (160), and submission of the release form (162) for approval for a build (170) of a software product. System (10) further provides for the permanent identification (171) of versions of source modules (160) with a specific build (170) of a product. The built load and other related files are thus labeled and stored in a build directory (540). A load list (550) includes a subset of laleled files in the build directory (540), which is referenced by a media downloader (532) adapted for copying files listed on the load list (550) from the build directory to the predetermined media (560), such as a tape (562) or compact disc (564).

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

SOFTWARE RELEASE MEDIA DOWNLOAD SYSTEM AND METHOD

TECHNICAL FIELD OF THE INVENTION

This invention is related in general to the field of computer software systems. More particularly, the invention is related to a software release media download system and method.

BACKGROUND OF THE INVENTION

Large-scale software development efforts require proper coordination and management of teams of software engineers and test engineers. When a software development effort involves a large group of engineers simultaneously working on multiple versions and releases of a large number of different source modules of the software product, confusion and inefficiency easily results if the development process and subsequent product release are not properly managed.

For example, some engineers may be coding source modules for version 3.1 of a software product X, while some engineers may be incorporating newer features into source modules for version 4.0, and still some engineers may be

providing a fix to some problems reported in source modules of version 2.1. Note that it is possible to have overlap between the three groups of engineers, so that an engineer may be involved in all three efforts.

Compounding the problem is the fact that each version of a software product must pass through multiple developmental stages prior to its release, where advancing to the next stage requires the passing of some predetermined testing and approval process. To be tested, all the source modules for that version of the software product must be collected and built into a load. The process of load building is also called compiling and linking of all the source modules. The resultant load or build is the software product to be tested or to be delivered to the customers when all the developmental stages have been passed.

Previously, the process of determining which source modules to collect, collecting the source modules, and building the load therefrom was tedious and time-consuming. Many hours were spent in meetings with software engineers and managers to determine which source modules should be included in which version, and what reported problems were fixed by which source modules, and whether the source modules with new features should be included in the present build. The end result is a list of source modules that is used to identify and pull the source modules, typically from a version control subsystem into which engineers have checked in their work. The source modules are then

manually identified and compiled into a load, and manually downloaded into a storage medium such as a tape or compact disc for delivery to the customers.

It may be seen that because the process of building a software load is repeatedly performed in the development of the software product, considerable savings in time, energy, and funds are possible if the process is automated and made more efficient.

SUMMARY OF THE INVENTION

Accordingly, there is a need for apparatus and method for file release media download that automates and stream lines the process.

In accordance with the present invention, apparatus and method for file release media download are provided which eliminate or substantially reduce the disadvantages associated with prior methods.

In one aspect of the invention, a load builder builds a load with checked-in and labeled source modules and stores it with related files in a build directory. The related files includes a label similar to that of the source modules. A load list includes a subset of files in the build directory, which is referenced by a media downloader adapted for copying files listed on the load list from the build directory to the predetermined media, such as a tape or compact disc.

In another aspect of the invention, a method for controlling software download to media provides the steps

of receiving and storing check-in data related to source modules and related files, including a load list, checked into a version control system, then tagging source modules and related files associated with the check-in data attached to the release form of the approved build with a unique build label. A load is built with source modules having the unique build label, and the load and related tagged files are copied to a build directory. The load is then downloaded with a subset of related files in the build directory to storage media in response to the load list.

BRIEF DESCRIPTION OF THE DRAWINGS

For a better understanding of the present invention, reference may be made to the accompanying drawings, in which:

FIGURE 1 is a simplified block diagram of an exemplary software release control system constructed according to the teachings of the present invention;

FIGURES 2A and 2B are an exemplary flowchart of a software release control process according to the teachings of the present invention;

FIGURE 3 is a block diagram illustrating the software release control process;

FIGURE 4 is a diagram illustrating the users of software release control system;

FIGURE 5 is an exemplary flowchart of a process to attach source modules to release forms;

FIGURE 6 is an exemplary flowchart of a process to assign release forms to a build;

FIGURE 7 is an exemplary flowchart of a create new product process;

FIGURE 8 is an exemplary flowchart of a process to create a build;

FIGURE 9 is an exemplary flowchart of a process for defect tracking;

FIGURE 10 is a block diagram illustrating defect tracking;

FIGURE 11 is a simplified block diagram of a remote access to software release control system; and

FIGURE 12 is a simplified block diagram of a media download system according to the teachings of the invention.

DETAILED DESCRIPTION OF THE INVENTION

The preferred embodiment(s) of the present invention is (are) illustrated in FIGURES 1-12, where like reference numerals being used to refer to like and corresponding parts of the various drawings.

Referring to FIGURE 1, a block diagram of a software release control system 10 constructed according to the teachings of the present invention is shown. Software release control system 10 uses a version control subsystem 12 to manage the numerous versions of software modules or files developed by software engineers. Version control subsystem 12 may be coupled to software release control

system 10 and a number of workstations, personal computers, terminals, or any suitable data display and entry device 14, via a computer network 16. At least two databases or files are included in version control subsystem 12 - source files 18 for storing source modules being developed or modified, and third party files 20 for storing source modules of third party software that will be incorporated into the load. During code development, engineers check out source files to work on and check them back in at the end of the work session. Version control subsystem 12 generally keeps track of what changes were made to the file. Examples of available version control subsystems include ClearCase by Pure Atria Software of Shaumburg, Illinois; Control Version System by Free Software Foundation of Cambridge, Massachusetts; DSEE by Hewlett Packard of Palo Alto, California.

Further coupled to software release control system 10 is another data display and entry device 22 provided for configuration administrator(s), who has authority to initiate and oversee the load building process. A problem reporting tool 24, represented in FIGURE 1 by a data display and entry device, is also coupled to software release control system 10. Problem reporting tool 24 is used to record problems reported by customers, test engineers, and other sources. Each problem report is assigned a unique problem report number for tracking and auditing purposes.

Software release control system 10 includes a number of tools, including a load builder 30 and a defect tracker 32. A number of files or databases are also included for storing a variety of data: check-in data 40, approved files 42, deferred files 44, submitted files 46, load list 50, and build report 52. Check-in data database 40 stores records associated with source modules that have been checked into version control subsystem 12. Check-in data 40 may include the developer's name, file name, check-in number, product, release, check-in time, total number of lines, number of lines changed, etc. Approved files database 42 stores data associated with source modules that have received approval for inclusion into a build, while deferred files database 44 stores data associated with source modules that have been denied inclusion into a build. Submitted files database 46 stores data associated with those source modules that have been attached to release forms. Release forms are logical groupings of source modules collected and submitted for the build process. Load list file 50 contains a list of built modules and third party software that have been identified to go onto deliverable media. The load list is used during generation of the deliverable media. Build report database 52 stores data generated from the load building process. Hard copy reports may then be generated from data stored in build report database 52.

After a load is built, it may be downloaded to a portable storage medium 60, such as a tape 62 and compact

disc 64. Storage medium 60 containing a load may then be tested on a test bed 66. In addition, the load may be electronically transferred to a remote site 68, via telecommunications networks or computer networks, for operations and/or testing.

FIGURES 2A and 2B is a flowchart describing the process of software release control 100. References may also be made to various system components shown in FIGURE 1 and to the diagram in FIGURE 3 providing an illustration of the process.

A user, typically a software engineer engaged in the development, modification, or testing of software code, logs into system 10 and selects a software product from a displayed list of existing or developing software products, as shown in block 102. If the source module that the engineer desires to work on is already checked into version control subsystem 12, then it is checked out therefrom, as shown in block 104. The engineer then codes or modifies the source module, as shown in block 106. At the end of the work session, the source module is checked back into version control subsystem 12 in block 108. When a source module is checked into version control subsystem 12, a trigger sends check-in data to software release control system 10, as shown in block 110. This check-in process is shown in FIGURE 3, where source modules 160 are checked into version control subsystem 12 and causing triggers to be invoked and received by software release control system 10.

In block 112 of the flowchart, the check-in data are stored by software release control system 10. If the source module is completed, then its associated check-in data may be attached to open release form 166 and 168, as shown in block 116 and illustrated in FIGURE 3. A release form is a logical grouping of check-in data associated with source modules checked into version control subsystem 12. A release form is typically associated with a particular problem report number or feature specification documentation number. When a release form is associated with a problem report number, the source modules associated therewith are developed or modified in response to the problem reported. When a release form is associated with a feature specification documentation number, the source modules associated therewith are typically developed for a new release. Once the release forms are complete, they are submitted as candidates for a particular build 170, as shown in block 118 and illustrated in FIGURE 3.

Software release control system 10 preferably provides a graphical and menu-driven user interface for prompting for data entry and displaying options. For example, to attach check-in data of source modules to a release form, the user may select from a list of possible functions related to release forms, such as listing of currently open release forms, creating a new release form, attaching check-in data to a release form, submitting a release form, etc. A pointing device such as a mouse may be used to select the desired option from the list. To attach check-

in data to release forms, the user may select the appropriate option, which brings up a list of currently open release forms for selection. The selection of a release form then causes a list of unattached check-in data for the software product in question that are associated with the particular user. The user may then select one or more check-in data for attachment to the release form. The user may also be prompted to provide additional data for each check-in data selected, such as the date of any preliminary logical inspection of the source module (such as a Fagan inspection), a problem report number or a feature specification documentation number, etc.

Returning to FIGURES 2A and 2B, in block 120, when release forms are submitted for a build, an electronic notification message may be automatically generated and sent to designated personnel having the responsibility of assigning and approving submitted release form(s) to a particular build, such as a configuration administrator. The configuration administrator may then assign the release form to a build and approve the build, as shown in block 122. The release forms are sorted according to approval - those release forms approved for a build and those release forms not approved for a build, as shown in blocks 124 and 126. The release forms not approved for a build may be deferred for a later build, and the associated data are stored, as shown in block 128. Although not shown explicitly, release forms may also be disapproved or rejected for a number of reasons, which may be unsubmitted

or deleted from system 10 after a predetermined time period. In block 130, the approved release forms are then provided in a list, which is used to tag or label all source modules 160 stored in database 18 that are to be included in build 170, as shown in block 132 and FIGURE 3. The build label identifies the product, version, and build in which the tagged source modules will be incorporated. System 10 further provides for the permanent identification 171 of versions of source modules 160 with a specific build 170 of a product.

A configuration administrator may initiate a build after all necessary release forms have been submitted, approved, assigned, and tagged. Software release control system 10 first retrieves source modules that bear the appropriate build label from version control subsystem 12, as shown in block 134, and also retrieves any third party software from version control subsystem 12, as shown in block 136. A build script is then invoked and executed to compile and link the source modules, third party software, and any other files necessary for the execution of the resultant software product, as shown in block 138. As shown in FIGURE 3, the built load may be in any one of development stages 176: development, integration test, system test, verification, and field deployment, and is so identified in its load number or part number identifier. For example, a load number may indicate, sequentially, the customer identifier, part number, release, point release, maintenance release, feature release, development state V,

and version number. Development state V may indicate the verification stage, for example. In block 142, the build may then be downloaded to a portable storage medium for delivery to a customer, electronically transferred to a desired destination for delivery or testing, or downloaded to a test bed for testing purposes. A build report summarizing information related to the build may also be generated.

It may be seen that software release control system 10 preferably provides a number of security levels to users having differing needs and assigned tasks. Referring to FIGURE 4, users 200 of system 10 may be assigned roles, such as system 202, administration 204, manager 206, and regular 208. For example, regular users 208 may have access to those functions related to creating release forms, attaching the source modules to release forms, and submitting release forms. Manager users 206 may have the additional ability to assign release forms to builds for those software products for which they have authorization during selected development stages, such as development and integration test. Administration users 204 may additionally have access to build administration functions, such as initiating the tagging source modules with build labels and specifying third party software. System users 202 may have unrestricted access to all resources.

FIGURE 5 provides additional details on an exemplary process to attach a source module to a release form 220. From a display of a list of open release forms, the user

may select one or more release forms, as shown in block 222. A listing of source modules associated with the user and checked into version control subsystem 12 are then displayed for the user's selection, as shown in block 224. For each selected source module, the user is prompted to provide certain predetermined information, such as whether a preliminary inspection, for example Fagan inspection, was performed and date of the inspection, as shown in block 226. The user will also be prompted to enter problem report number(s) or feature specification document number(s) associated with the source module, as shown in block 228. Software release control system 10 then groups the selected source module with the selected release form. The attachment process ends in block 230.

As described above, after source modules are attached to release forms primarily by the engineers or developers, the release forms are then assigned to a build, primarily by build administrators. Referring to FIGURE 6, an exemplary assignment process 240 is shown. A list of release forms submitted for a given product is displayed, as shown in block 242. The user, such as build administrator, selects a release form from the list, as shown in block 244. In block 246, details of check-in data associated with the selected release form are then displayed, so that the user may review the details and indicate his/her approval, as shown in block 248. If the user disapproves of the release form, then an electronic message may be generated and sent to the developer(s)

associated with the source modules so that appropriate action may be taken, as shown in block 250. If the release forms are approved, then a list of builds for the selected software product is displayed, as shown in block 254. The user may then select the build that will incorporate the approved release form, as shown in block 256. The process ends in block 258 thereafter.

Referring to FIGURE 7, an exemplary process 270 for creating a new software product in software release control system 10 is shown. As described above, it is preferable to limit access to this function to users with system authority. When this option is selected, system 10 prompts for the product identifier and release number, as shown in block 272. A sequence number and a build index number are then set by system 10, as shown in blocks 274 and 276. For example, sequence number may be set to 1 and the build index number may be set to 0. The process ends in block 278. When the user enters product and release number of an existing product, error messages may be displayed.

FIGURE 8 shows an exemplary process for creating a new build 290. When this function is selected by the user, the screen displays a prompt for the user to identify the owner of the build, as shown in block 292. system 10 then queries and obtains from the check-in database information on software products and previous builds known to be related to the owner identifier entered by the user, as shown in block 294. The result is displayed for the user to select the software product and a build, as shown in

block 296. Dependencies such as operating system and environment variables which specify certain values to be used in the build are either copied from an existing build and modified or entered by the user for the new build, as shown in block 298. Other information such as specifying third party software may also be entered at this time. A build label for the new build is then generated, based on the information given, as shown in block 300. The create new build process ends in block 302.

As described above, software release control system 10 is also capable of tracking defects in previous versions and how these defects are fixed in subsequent versions. An exemplary process 320 performed by defect tracker 32 (FIGURE 1) is shown in FIGURE 9. References are also made to FIGURE 10, which provides a graphical illustration of the process. A product and a build are first specified or selected from lists displayed by system 10, as shown in blocks 322 and 324. With this information, defect tracker 32 obtains a list of all check-in data of all source modules associated with the selected build and product, as shown in block 326. Problem report numbers that are specified in the check-in data are then obtained, as shown in block 328. As identified by the problem report numbers, those problem reports stored in problem reporting tool 24 (FIGURE 1) are labeled with the build label of the present build, as shown in block 330.

It may be seen in FIGURE 10 that source module check-in data 360 and 362, a release form 364 that includes

check-in data 360 and 362, and load 366 are all tagged with a build label 370-376. In determining which check-in data and correspondingly, which source module contains the code for fixing a known defect, a problem report number 384 associated with check-in data 360 is used to identify a problem report 368 with the problem report number identifier 380. Problem report 368 is then tagged with a build label 388 that corresponds to load 366. Therefore, the proper association between a load and problem report(s) is made.

FIGURE 11 shows that a software release control system 400 may also be accessed from a site 410 located remotely from system 400. Software release control system 400 includes a load builder 401, a defect tracker 402, and a database 403, which stores a number of files as described above in conjunction with FIGURE 1. A version control subsystem 404 having a database storing source files 405 is coupled to system 400. A workstation 406 may also be coupled to system 400 to provide administrator access thereto. At remote site 410, a second version control subsystem 411 storing source files 412 generated and/or maintained by software engineers at remote site 410 is provided. Version control subsystem 411 is coupled to developer workstations, personal computers, and other suitable tools 416 which facilitate code development.

At the end of a work session, a software engineer checks in source modules to remote version control subsystem 411, which are then stored in source files

database 412. As the source modules are checked in, a trigger is invoked and sent to software release control system 400. Similar to the local site application as shown in FIGURE 1, the trigger includes check-in data. The trigger may be transmitted over telecommunications networks, computer networks, or a dedicated line, as deemed suitable. In a periodic manner, packets containing the contents of source files database 412 are electronically copied to source files database 405 of local version control subsystem 404 to synchronize the databases, so that the contents thereof are essentially identical over time.

When a user has completed the source modules, he/she may attach them to one or more release forms and then submit the release forms for a build, in the same manner as described above. When release forms are approved for a build, the corresponding source modules stored in remote source files database 412 that make up the release form are tagged with the appropriate build label. The load building process obtains approved files or source modules from remote version control subsystem 411 to build the load. Defect tracking for the remote site is performed in a similar manner as described above, where a build label becomes associated with problem reports. Constructed in this manner, developers may submit source files from one or more remote sites to one local software release control system for load building and defect tracking.

Referring to FIGURE 12, a media download control subsystem 510 of the file release control system is shown.

System 510 includes a load builder 530 described in detail above, and additionally a media downloader 532. Compiled and linked source modules and other files generated during the load building process are stored in a build directory 540 at the end of the load building process. Build directory 540 functions as a staging area for downloading the files to deliverable storage media 560, such as tapes 560 and compact discs 564. System 510 further includes checksum file(s) 542, release level file(s) 544, which are generated during the media downloading process.

Also generated as part of the software development effort and stored in version control subsystem 12 is a load list 550. Load list 550 enumerates all files that are to be downloaded onto the deliverable media during the downloading process, which is a subset of files residing in build directory 540. When the software is to be downloaded onto more than one compact disc or tape, multiple load lists may be used. Typically, load list 550 is specific to a version, release, and customer.

In operation, during the load building process described above, source modules for a specific version and release are compiled and linked. The compiled and linked executable source files are then stored in build directory 540. The load building process also copies other related files to building area, including load list 50 and release level files and information 544 such as software product part number (version and release number) data, installation scripts, third party software, data/configuration files,

license files, libraries, etc. associated with the current build. These files are similarly controlled in version control subsystem 12 and similarly tagged with the same or similar build label as the source modules to associate files related to the same version and release as shown in FIGURES 3, 8, and 10. During the media downloading process, media downloader 532 uses load list 550 and verifies all files enumerated therein are in build directory 540. Media downloader 532 then computes a checksum for each file listed on load list 550, and generates a program that will compute the checksum of the files during customer installation and check the computed installation checksum against the checksum computed at download time. Media downloader 532 then copies all files listed on load list 550 and checksum files 542 to a target file, which is used to cut the compact disc or tape. A log 546 recording what files were downloaded onto the media is generated and stored into version control subsystem 12.

Constructed and automated in this manner, media downloading is a process integrated with file release control which uses build labels to associate all files related to the same version and release.

Although the present invention and its advantages have been described in detail, it should be understood that various changes, substitutions and alterations can be made therein without departing from the spirit and scope of the invention as defined by the appended claims.

WHAT IS CLAIMED IS:

1. A software media download system, comprising:

a load builder adapted for labeling each file related to a specific version and release of a software product with a predetermined label and copying the labeled files to a build directory;

a load list having the predetermined label and including a list of a subset of labeled files in the build directory;

a media downloader adapted for copying the subset of labeled files from the build directory to the predetermined media in response to the load list.

2. The system, as set forth in claim 1, wherein the subset of labeled files comprises a checksum program generated by the media downloader adapted for recomputing and comparing checksums of files during a software installation process.

3. The system, as set forth in claim 1, wherein the subset of labeled files includes a file having a part number of the software product.

4. The system, as set forth in claim 1, wherein the subset of labeled files includes an installation script file.

5. The system, as set forth in claim 1, wherein the subset of labeled files includes third party software.

6. The system, as set forth in claim 1, wherein the subset of labeled files includes configuration files.

7. The system, as set forth in claim 1, wherein the media downloader further comprises a process for generating checksums of files downloaded to the media and copying the checksums to the media.

8. The system, as set forth in claim 1, wherein the media downloader further comprises a process for generating a checksum program for recomputing and comparing the checksums of the files offloaded from the media upon installation and copying the checksum program to the media.

9. A method for controlling software download to media, comprising the steps of:

receiving and storing check-in data related to source modules and related files, including a load list, checked into a version control system;

tagging source modules and related files associated with the check-in data attached to the release form of the approved build with a unique build label;

building a load with source modules having the unique build label;

copying the load and related tagged files to a build directory; and

downloading the load and a subset of related tagged files in the build directory to storage media in response to the load list.

10. The method, as set forth in claim 9, further comprising the steps of:

computing checksums of the load and subset of related tagged files downloaded to the storage media;

generating a checksum program adapted for recomputing checksums of files installed onto a customer computer during installation and comparing these checksums with the checksums computed during media download; and

downloading the checksum program to the storage media.

11. The method, as set forth in claim 9, further comprising the step of verifying the subset of tagged files listed on the load list reside in the build directory.

12. The method, as set forth in claim 9, further comprising the step of recording the identity of all files successfully downloaded onto the storage media in a log file.

13. The method, as set forth in claim 9, wherein the downloading step comprises the step of downloading to at least one tape.

14. The method, as set forth in claim 9, wherein the downloading step comprises the step of downloading to at least one compact disc.

15. A system for software release with media download, comprising:

a load builder adapted for building a load of software source modules a software product, labeling the load and files related to a specific version and release of the software product with a predetermined label, and copying the files built from the labeled files to a build directory;

a load list having a list of files including a subset of files in the build directory; and

a media downloader adapted for copying files listed on the load list from the build directory to the predetermined media.

16. The system, as set forth in claim 15, wherein the subset of labeled files comprises a checksum program generated by the media downloader adapted for recomputing and comparing checksums of files during a software installation process.

17. The system, as set forth in claim 15, wherein the subset of labeled files includes a file having a part number of the software product.

18. The system, as set forth in claim 15, wherein the subset of labeled files includes an installation script file.

19. The system, as set forth in claim 15, wherein the media downloader further comprises a process for generating checksums of labeled files downloaded to the media and copying the checksums to the media.

20. The system, as set forth in claim 15, wherein the media downloader further comprises a process for generating a checksum program for computing the checksums of the files downloaded to the media and copying the checksum program to the media.

FIG. 1

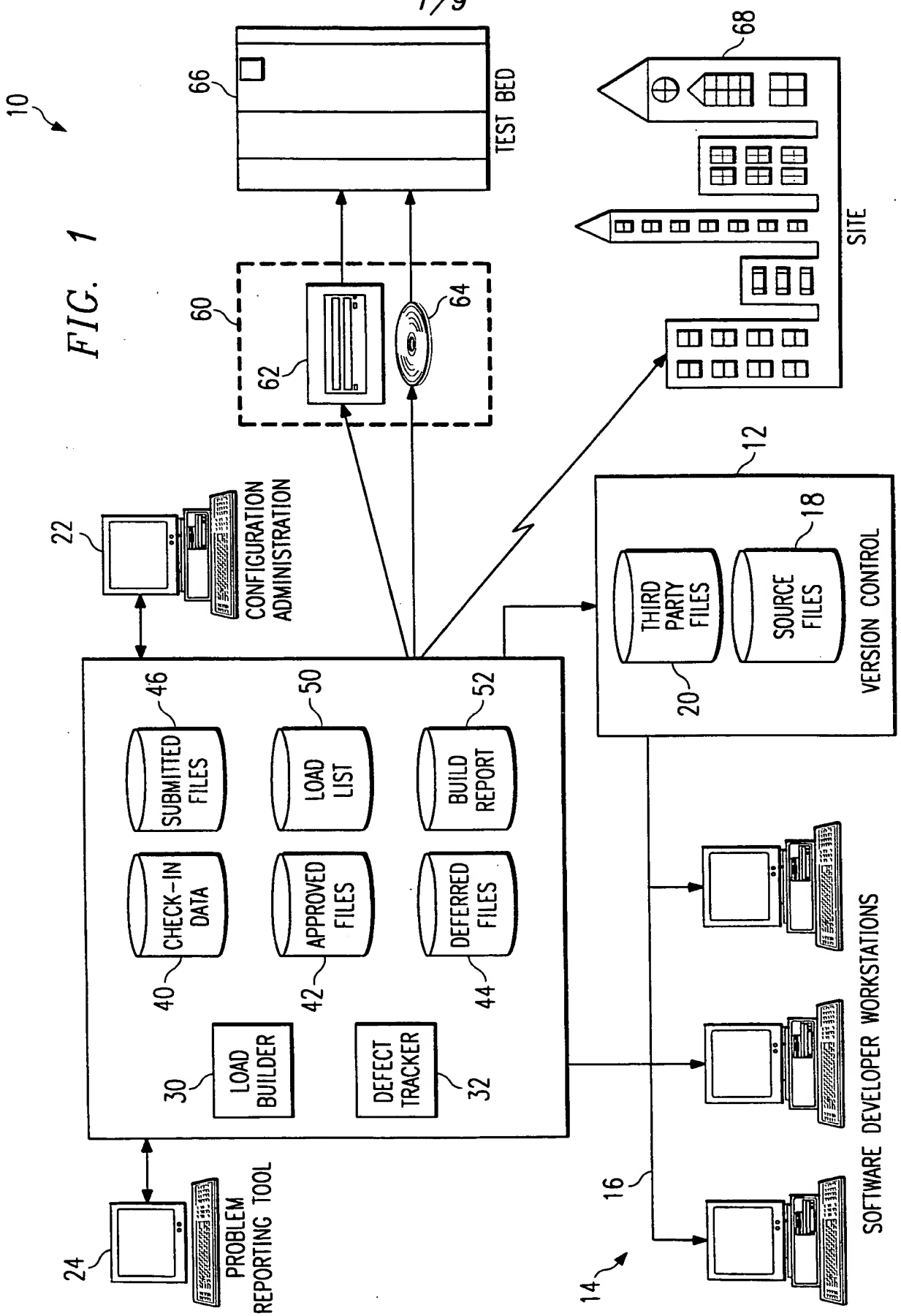

10
66
TEST BED
68
SITE
60
62
64
CONFIGURATION ADMINISTRATION
22
12
VERSION CONTROL
18
THIRD PARTY FILES
SOURCE FILES
20
46
SUBMITTED FILES
50
LOAD LIST
52
BUILD REPORT
CHECK-IN DATA
APPROVED FILES
DEFERRED FILES
40
42
44
30
LOAD BUILDER
DEFECT TRACKER
32
24
PROBLEM REPORTING TOOL
16
14
SOFTWARE DEVELOPER WORKSTATIONS

FIG. 2A

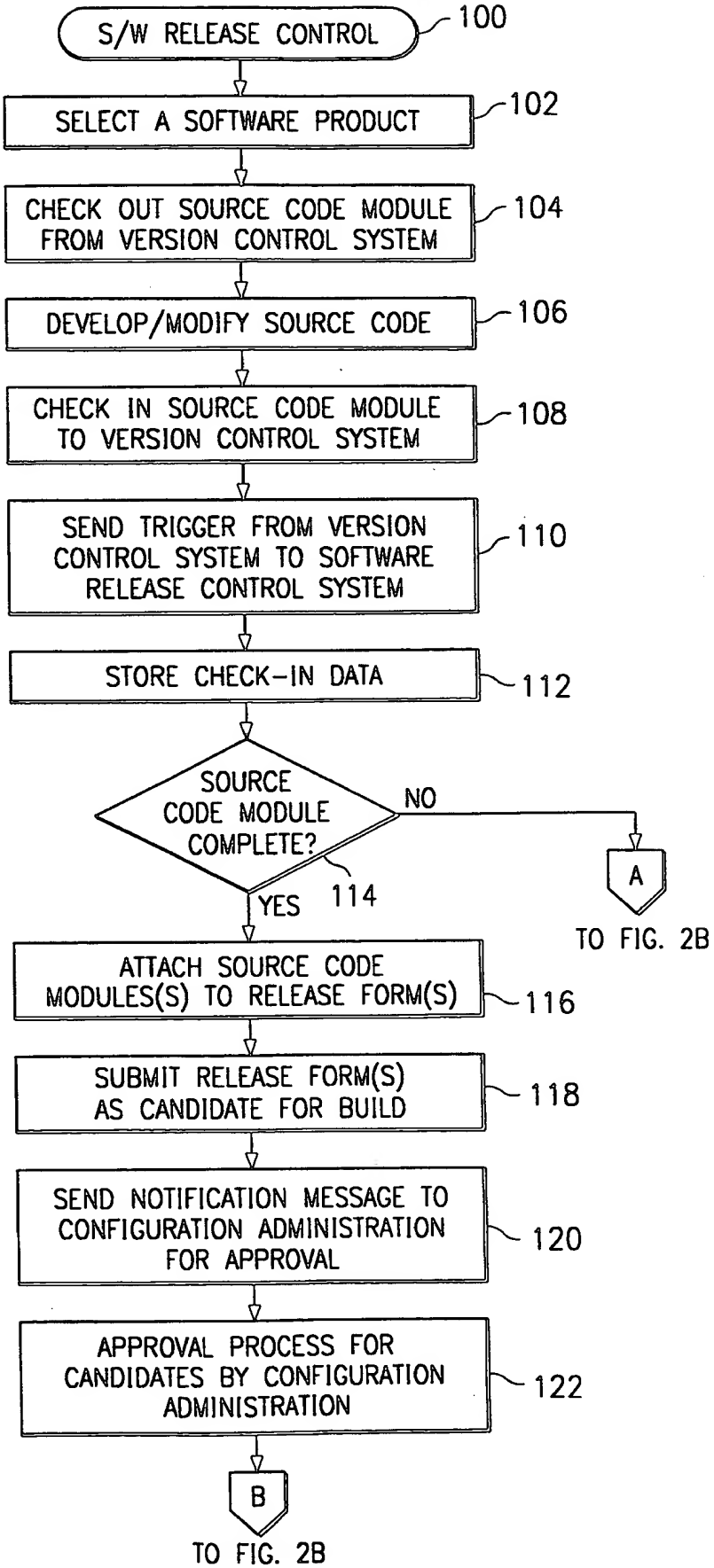

S/W RELEASE CONTROL
100
SELECT A SOFTWARE PRODUCT
102
CHECK OUT SOURCE CODE MODULE FROM VERSION CONTROL SYSTEM
104
DEVELOP/MODIFY SOURCE CODE
106
CHECK IN SOURCE CODE MODULE TO VERSION CONTROL SYSTEM
108
SEND TRIGGER FROM VERSION CONTROL SYSTEM TO SOFTWARE RELEASE CONTROL SYSTEM
110
STORE CHECK-IN DATA
112
SOURCE CODE MODULE COMPLETE?
114
NO
A
TO FIG. 2B
YES
ATTACH SOURCE CODE MODULES(S) TO RELEASE FORM(S)
116
SUBMIT RELEASE FORM(S) AS CANDIDATE FOR BUILD
118
SEND NOTIFICATION MESSAGE TO CONFIGURATION ADMINISTRATION FOR APPROVAL
120
APPROVAL PROCESS FOR CANDIDATES BY CONFIGURATION ADMINISTRATION
122
B
TO FIG. 2B

FIG. 2B

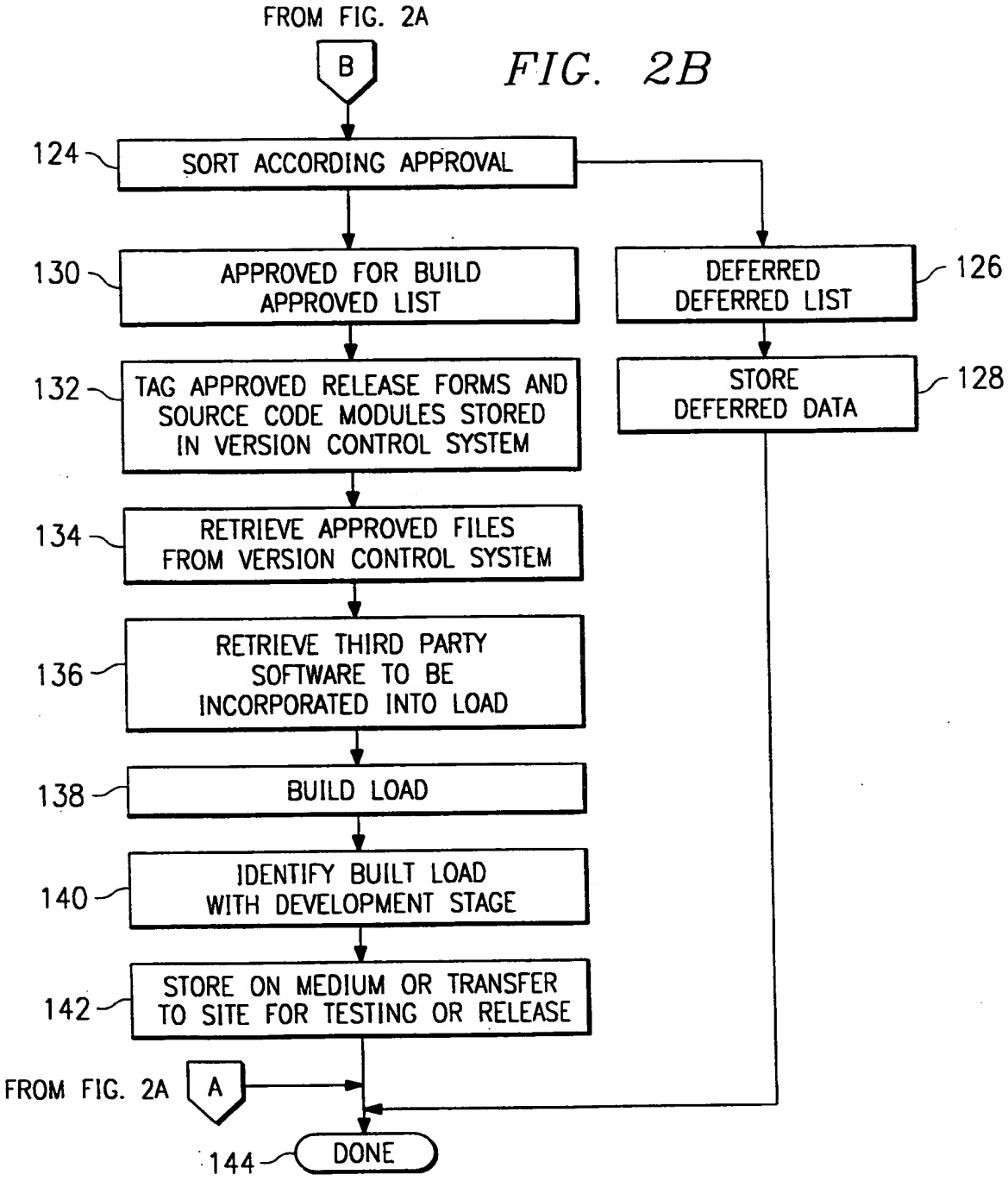

FROM FIG. 2A
B
124
SORT ACCORDING APPROVAL
130
APPROVED FOR BUILD
APPROVED LIST
DEFERRED
DEFERRED LIST
126
132
TAG APPROVED RELEASE FORMS AND
SOURCE CODE MODULES STORED
IN VERSION CONTROL SYSTEM
STORE
DEFERRED DATA
128
134
RETRIEVE APPROVED FILES
FROM VERSION CONTROL SYSTEM
136
RETRIEVE THIRD PARTY
SOFTWARE TO BE
INCORPORATED INTO LOAD
138
BUILD LOAD
140
IDENTIFY BUILT LOAD
WITH DEVELOPMENT STAGE
142
STORE ON MEDIUM OR TRANSFER
TO SITE FOR TESTING OR RELEASE
FROM FIG. 2A
A
144
DONE

FIG. 4

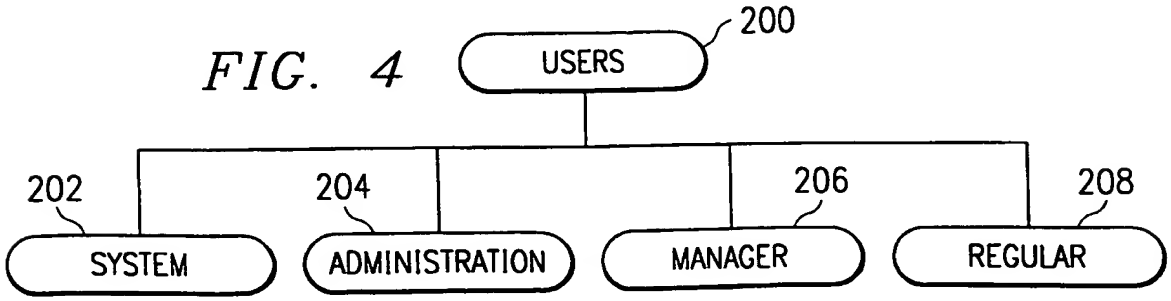
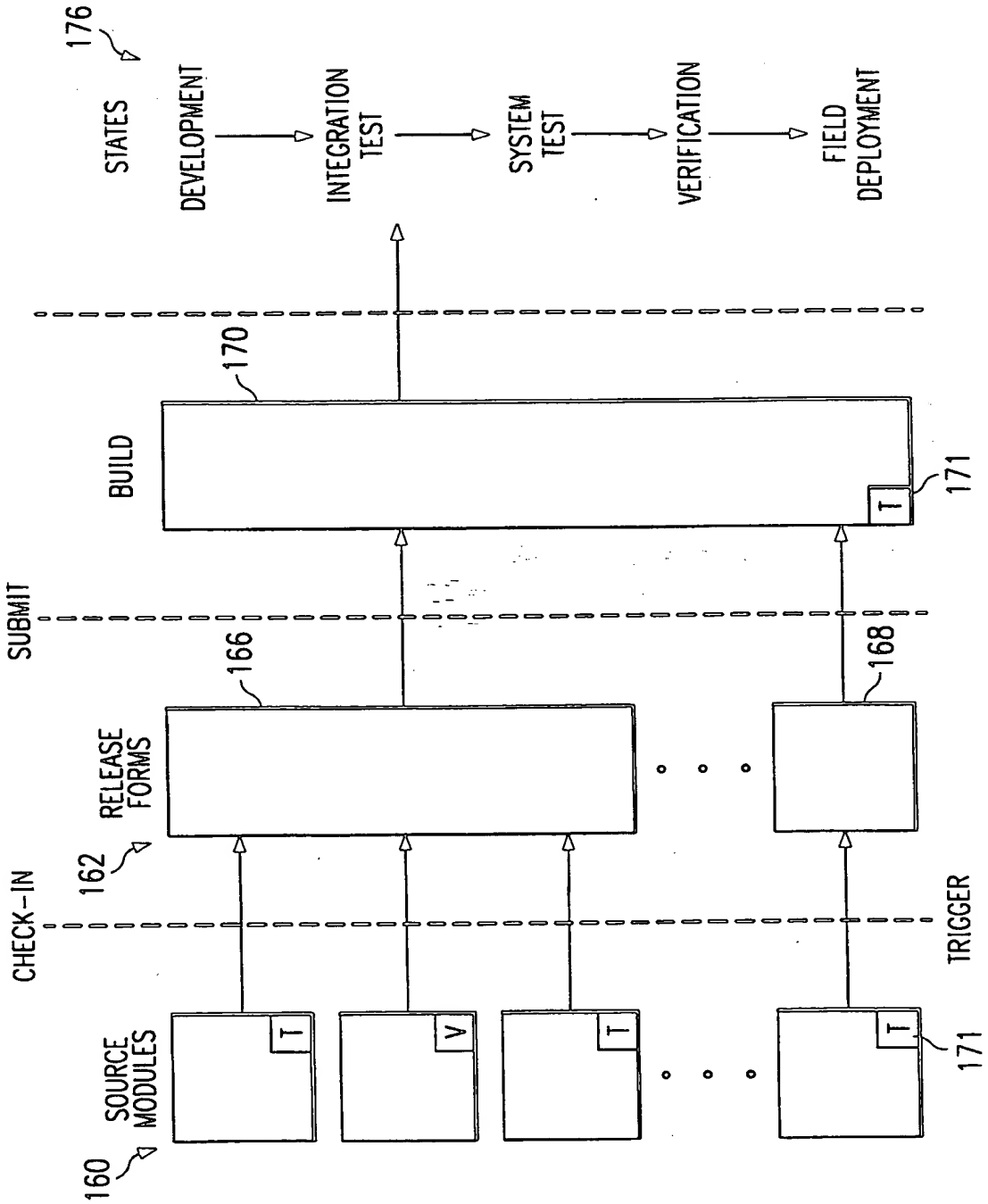

200
USERS
202
SYSTEM
204
ADMINISTRATION
206
MANAGER
208
REGULAR

FIG. 3


176
STATES
DEVELOPMENT
INTEGRATION TEST
SYSTEM TEST
VERIFICATION
FIELD DEPLOYMENT
170
BUILD
T
171
SUBMIT
166
168
RELEASE FORMS
162
CHECK-IN
TRIGGER
SOURCE MODULES
160
T
V
T
T
171

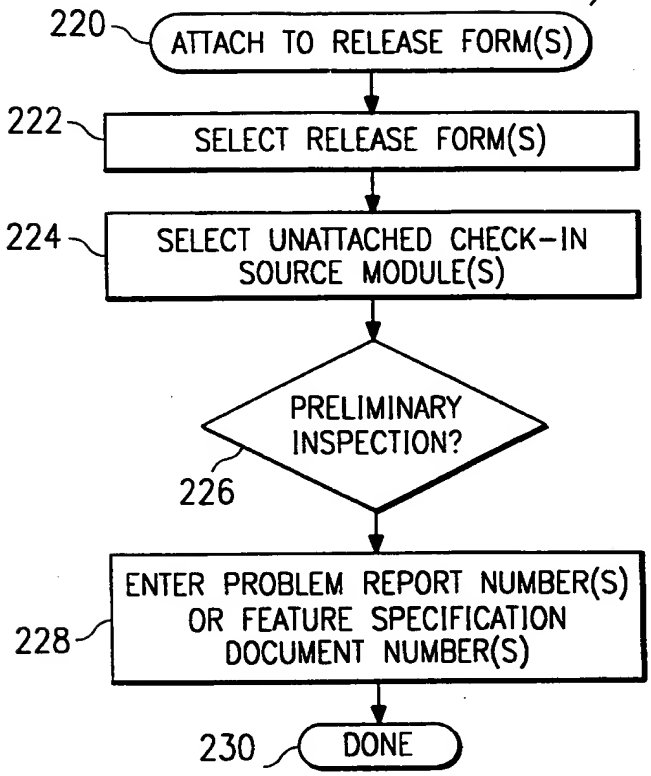

220
ATTACH TO RELEASE FORM(S)
222
SELECT RELEASE FORM(S)
224
SELECT UNATTACHED CHECK-IN SOURCE MODULE(S)
PRELIMINARY INSPECTION?
226
ENTER PROBLEM REPORT NUMBER(S) OR FEATURE SPECIFICATION DOCUMENT NUMBER(S)
228
230
DONE

FIG. 5

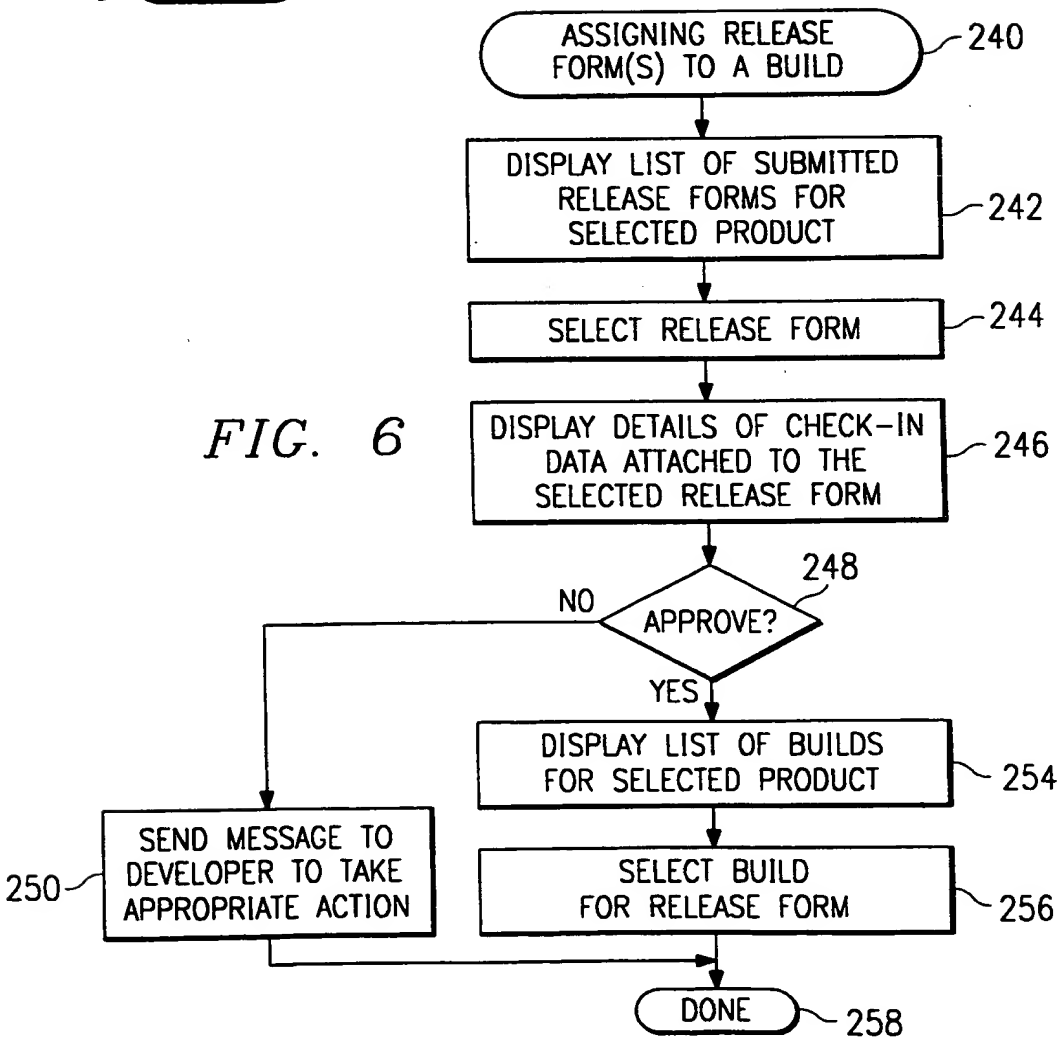

ASSIGNING RELEASE FORM(S) TO A BUILD
240
DISPLAY LIST OF SUBMITTED RELEASE FORMS FOR SELECTED PRODUCT
242
SELECT RELEASE FORM
244
DISPLAY DETAILS OF CHECK-IN DATA ATTACHED TO THE SELECTED RELEASE FORM
246
248
NO
APPROVE?
YES
DISPLAY LIST OF BUILDS FOR SELECTED PRODUCT
254
SEND MESSAGE TO DEVELOPER TO TAKE APPROPRIATE ACTION
250
SELECT BUILD FOR RELEASE FORM
256
DONE
258

FIG. 6

FIG. 7

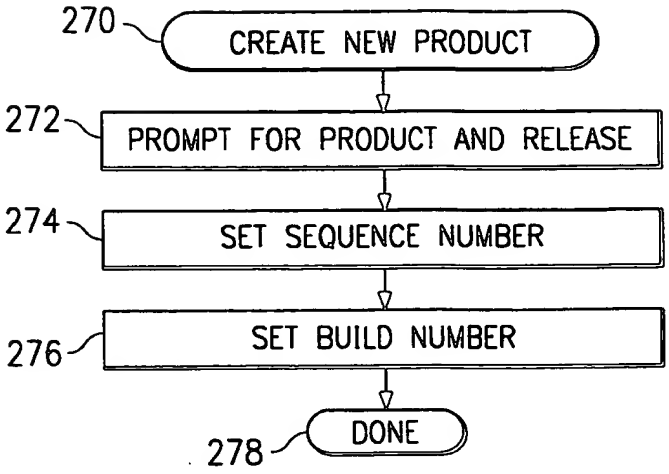
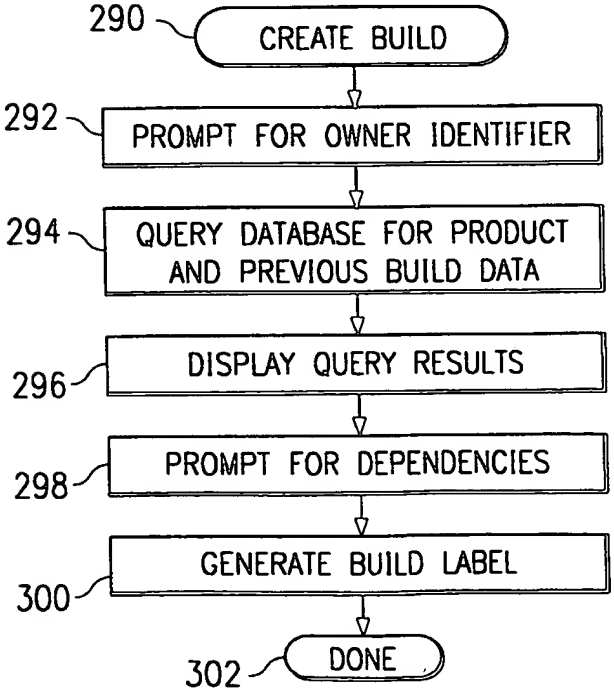

270
CREATE NEW PRODUCT
272
PROMPT FOR PRODUCT AND RELEASE
274
SET SEQUENCE NUMBER
276
SET BUILD NUMBER
278
DONE

FIG. 8


290
CREATE BUILD
292
PROMPT FOR OWNER IDENTIFIER
294
QUERY DATABASE FOR PRODUCT AND PREVIOUS BUILD DATA
296
DISPLAY QUERY RESULTS
298
PROMPT FOR DEPENDENCIES
300
GENERATE BUILD LABEL
302
DONE

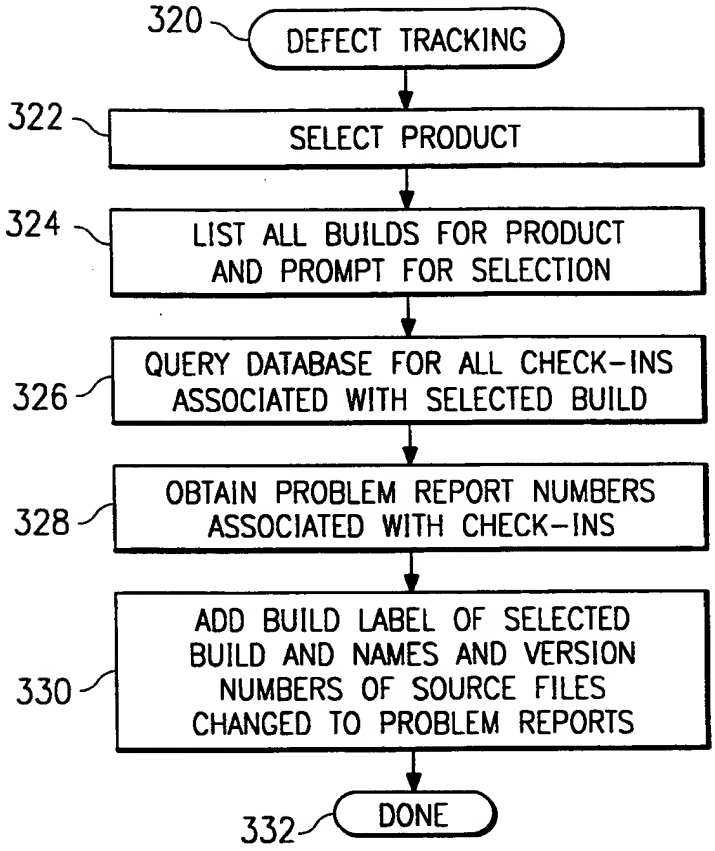

320
DEFECT TRACKING
322
SELECT PRODUCT
324
LIST ALL BUILDS FOR PRODUCT AND PROMPT FOR SELECTION
326
QUERY DATABASE FOR ALL CHECK-INS ASSOCIATED WITH SELECTED BUILD
328
OBTAIN PROBLEM REPORT NUMBERS ASSOCIATED WITH CHECK-INS
330
ADD BUILD LABEL OF SELECTED BUILD AND NAMES AND VERSION NUMBERS OF SOURCE FILES CHANGED TO PROBLEM REPORTS
332
DONE

FIG. 9

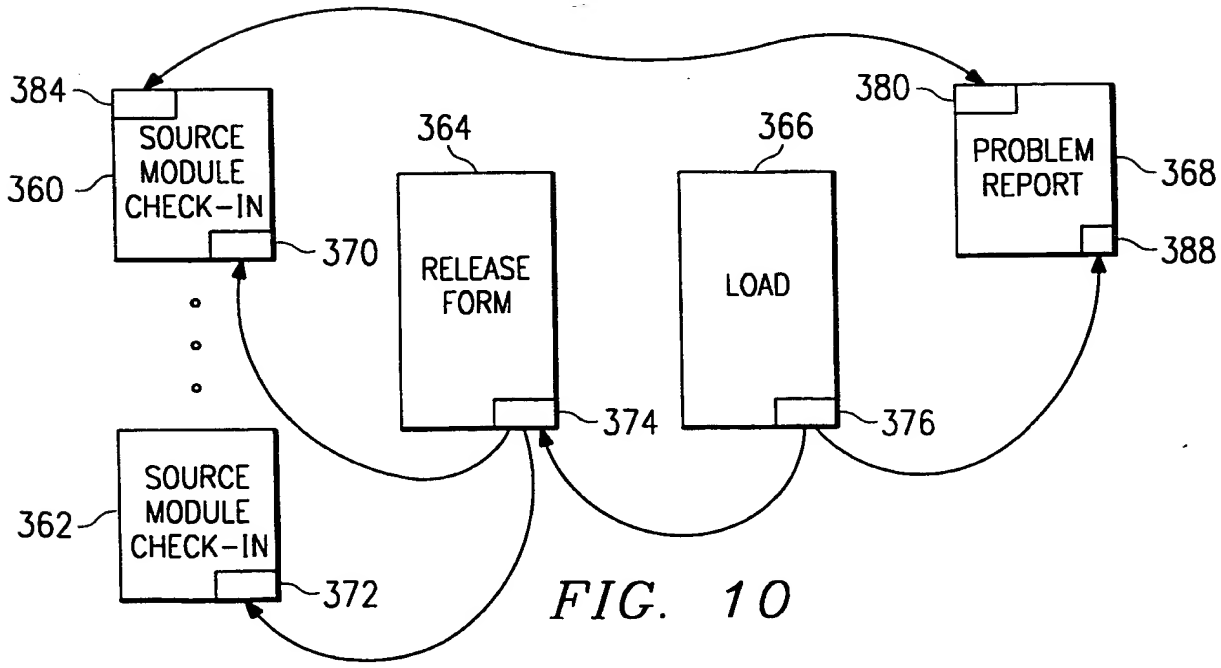

384
SOURCE MODULE CHECK-IN
360
370
364
RELEASE FORM
374
366
LOAD
376
380
PROBLEM REPORT
368
388
362
SOURCE MODULE CHECK-IN
372

FIG. 10

FIG. 11

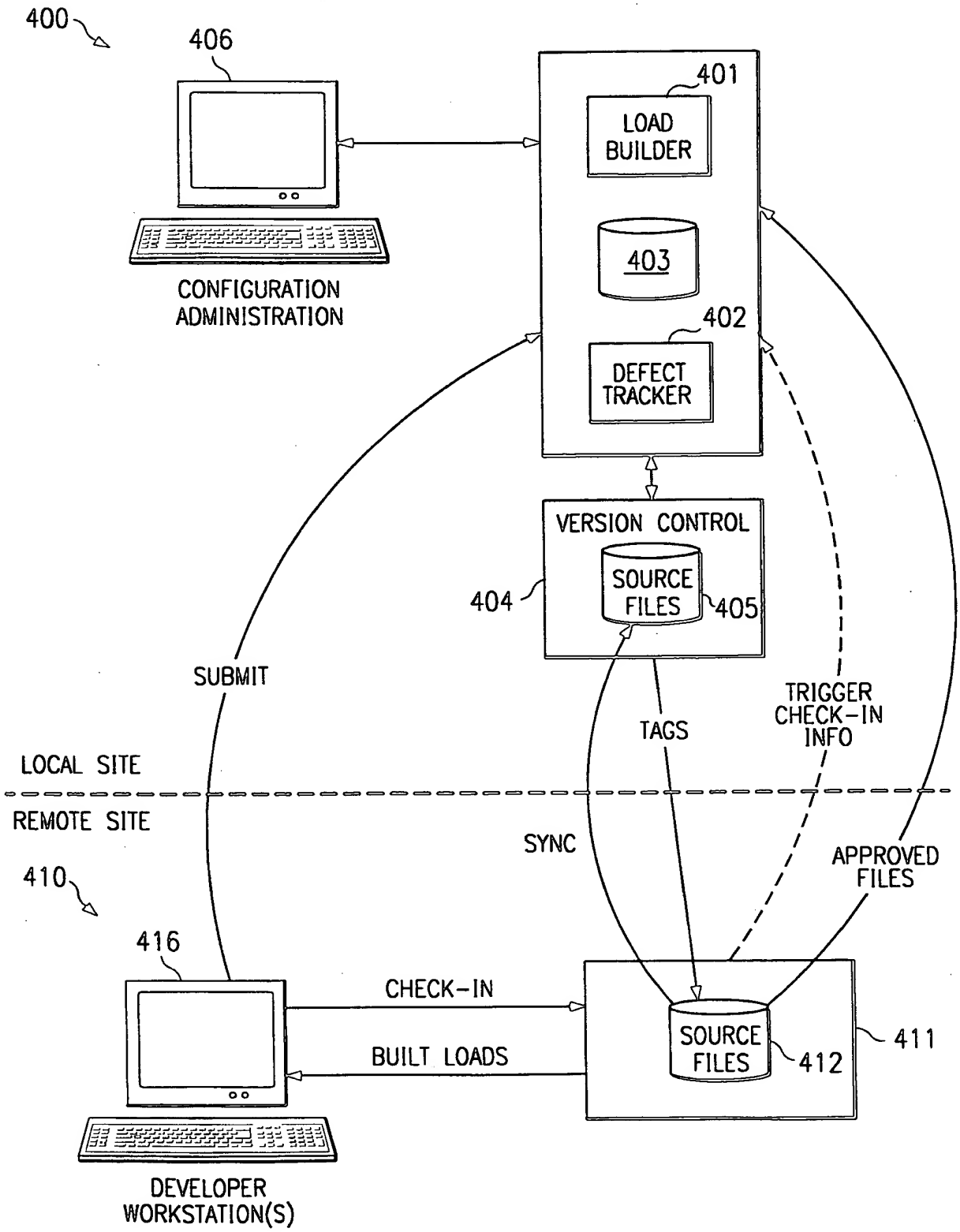
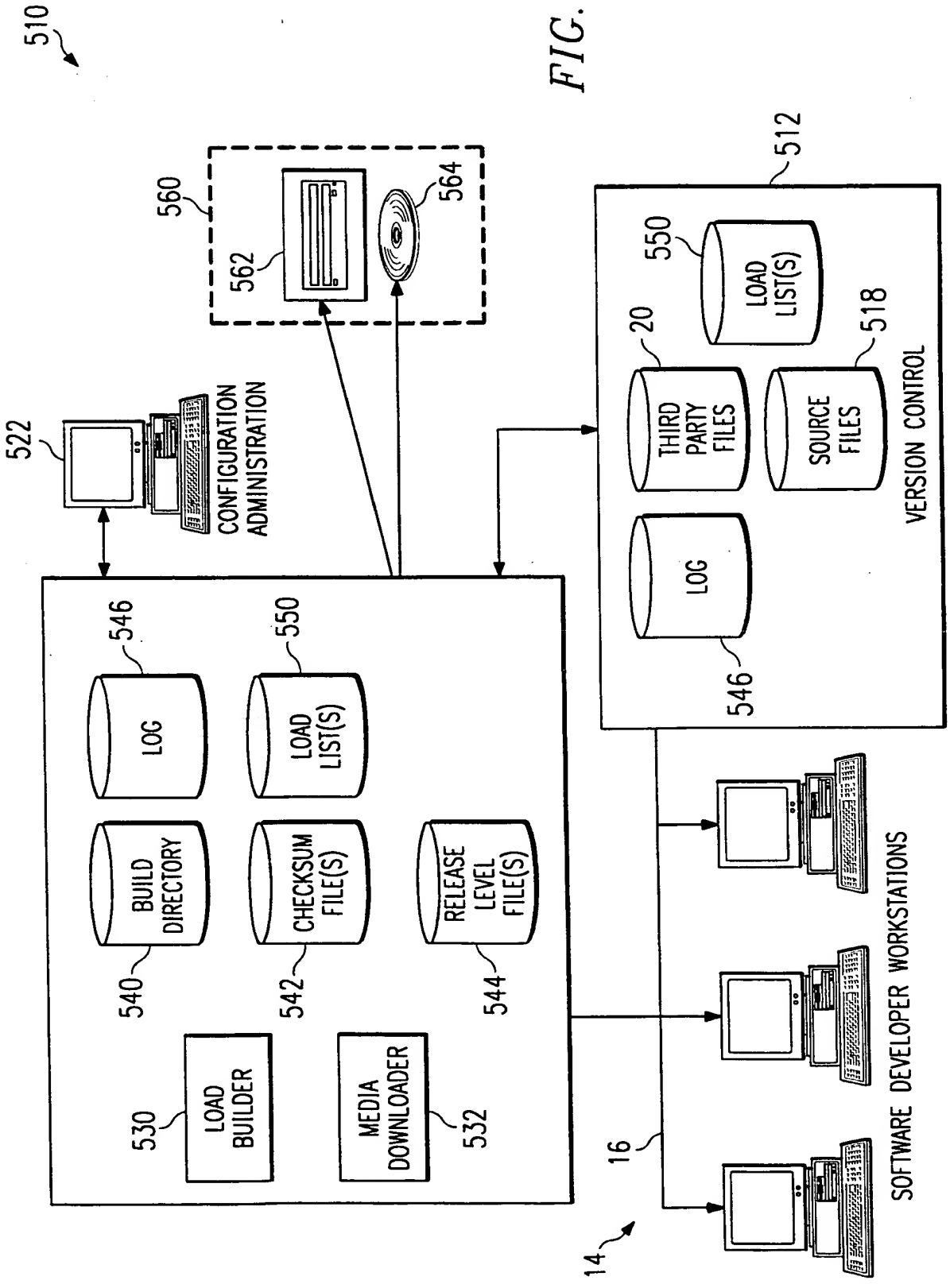

400
406
401
LOAD BUILDER
403
402
DEFECT TRACKER
CONFIGURATION ADMINISTRATION
VERSION CONTROL
SOURCE FILES
404
405
SUBMIT
TAGS
TRIGGER CHECK-IN INFO
LOCAL SITE
REMOTE SITE
SYNC
APPROVED FILES
410
416
CHECK-IN
BUILT LOADS
SOURCE FILES
412
411
DEVELOPER WORKSTATION(S)

FIG. 12



INTERNATIONAL SEARCH REPORT

Intern. al Application No
PCT/US 97/20396

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F9/44 G06F9/445

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 707 264 A (SUN MICROSYSTEMS INC) 17 April 1996 see page 3, line 18 - line 25 see page 3, line 50 - line 57 see page 6, line 5 - line 36 ---	1-3,5-7, 15-17,19
A	"PACKAGING TOOLS FOR THE ADE BUILD ENVIRONMENT" IBM TECHNICAL DISCLOSURE BULLETIN, vol. 37, no. 9, 1 September 1994, page 41 XP000473324 ---	1-20
A	"SOFTWARE PACKAGING AND VERIFICATION AID" IBM TECHNICAL DISCLOSURE BULLETIN, vol. 36, no. 6A, 1 June 1993, pages 223-225, XP000372411 see the whole document ---	1-20

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

1

Date of the actual completion of the international search	Date of mailing of the international search report
13 February 1998	20/02/1998

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl. Fax: (+31-70) 340-3016	Brandt, J

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Intern. .al Application No
PCT/US 97/20396

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	"DESIGN OF THE COMMIT INSTALL PLAN PROCESS FOR A NETWORK INSTALLATION PROGRAM" IBM TECHNICAL DISCLOSURE BULLETIN, vol. 38, no. 7, July 1995, page 357/358 XP000521719 see page 358, line 19 - line 24	4,18

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

Intern. .al Application No
PCT/US 97/20396

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0707264 A	17-04-96	US 5617533 A	01-04-97
		JP 8234966 A	13-09-96